Efficient Verifiable Computation of XOR for Biometric Authentication

Aysajan Abidin[1], Abdelrahaman Aly[1], Enrique Argones Rúa[1], Aikaterini Mitrokotsa[2]

[1]KU Leuven, ESAT/COSIC, Belgium and iMinds, Belgium
Firstname.Lastname@esat.kuleuven.be
[2]Chalmers University of Technology, Gothenburg, Sweden
aikmitr@chalmers.se

Abstract. This work addresses the security and privacy issues in remote biometric authentication by proposing an efficient mechanism to verify the correctness of the outsourced computation in such protocols. In particular, we propose an efficient verifiable computation of XORing encrypted messages using an XOR linear message authentication code (MAC) and we employ the proposed scheme to build a biometric authentication protocol. The proposed authentication protocol is both secure and privacy-preserving against *malicious* (as opposed to *honest-but-curious*) adversaries. Specifically, the use of the verifiable computation scheme together with an homomorphic encryption protects the privacy of biometric templates against malicious adversaries. Furthermore, in order to achieve unlinkability of authentication attempts, while keeping a low communication overhead, we show how to apply Oblivious RAM and biohashing to our protocol. We also provide a proof of security for the proposed solution. Our simulation results show that the proposed authentication protocol is efficient.

Key words: Verifiable computation, universal hash functions, homomorphic encryption, biometric authentication, template privacy and security.

1 Introduction

Following the rapid growth of mobile and cloud computing, outsourcing computations to the cloud has increasingly become more attractive. Many practical applications, however, require not only the privacy of the sensitive data in such computations, but also the verifiability of correctness of the outsourced computations. There has been a wealth of work on verifiable computations in recent years, see, e.g., [1–3] and the references therein. One type of outsourced computation, in biometric authentication with distributed entities, is the computation over encrypted bitstrings (e.g., encrypted biometric templates) to obtain the XOR of two bitstrings

(e.g., the XOR of the fresh and reference biometric templates). Consider, for instance, the following biometric authentication protocol consisting of three entities, namely, a set $\mathcal{C}$ of clients $\mathcal{C}_i$, for $i = 1, \cdots, N$, one for each user $\mathcal{U}_i$, a cloud server $\mathcal{CS}$ with a database $\mathcal{DB}$, and an authentication server $\mathcal{SP}$. Each client $\mathcal{C}_i$ has a sensor that extracts biometric templates from its owner's biometrics (e.g., fingerprints). The cloud server $\mathcal{CS}$ stores the reference biometric templates and performs calculations. The authentication server $\mathcal{SP}$ takes the final decision depending on whether there is a match between the fresh and the reference biometric templates. This is a reasonable model adopted in many research papers (cf. Related Work) and the industry (e.g., [4]) considering the fast rise of cloud computing and storage services, and also the widespread use of smartphones with embedded biometric sensors. However, the privacy of biometric features must be seriously taken into account in such architectures, since its disclosure may lead to breaches in security and traceability of users among services, besides the inherent private information disclosure.

Let us consider a simple example of a biometric authentication protocol using an homomorphic encryption scheme. Let $\mathsf{HE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a hypothetical homomorphic encryption (HE) scheme and f a function such that $f\big(\mathsf{Enc}(m), \mathsf{Enc}(m')\big) = \mathsf{Enc}\big(m \oplus m'\big)$, for m, m' in the domain of Enc, where $\oplus$ is the XOR operation. Suppose that the encryption/decryption keys pk/sk are generated by the authentication server $\mathcal{SP}$ and pk is distributed to $\mathcal{CS}$ and all $\mathcal{C}_i$. Then, the protocol works as follows. During the *enrollment phase*, the client $\mathcal{C}_i$ provides an encrypted reference biometric template $\mathsf{Enc}(b_i)$, along with the user ID_i for storage in the database $\mathcal{DB}$ on the $\mathcal{CS}$ side. During the *authentication phase*, the client $\mathcal{C}_i$ provides an encrypted fresh biometric template $\mathsf{Enc}(b'_i)$ and a claimed user ID_i to $\mathcal{CS}$, which then retrieves $\mathsf{Enc}(b_i)$ corresponding to ID_i from its database, computes $\mathsf{ct}_{b_i \oplus b'_i} = f\big(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i)\big) = \mathsf{Enc}\big(b_i \oplus b'_i\big)$ and sends $\mathsf{ct}_{b_i \oplus b'_i}$ to $\mathcal{SP}$. Finally, $\mathcal{SP}$ decrypts $\mathsf{ct}_{b_i \oplus b'_i}$ and checks if the Hamming weight $\mathsf{HW}(b_i \oplus b_i) \leqslant \tau$, where τ is a predefined authentication threshold. If $\mathsf{HW}(b_i \oplus b_i) \leqslant \tau$, then the user is granted access; otherwise, he/she is rejected. Note that $\mathsf{HW}(b_i \oplus b'_i)$ is equal to the Hamming distance $\mathsf{HD}(b_i, b'_i)$.

At a first glance, the protocol may seem secure against a malicious $\mathcal{CS}$, with respect to both the fresh and the stored template privacy. However, this only holds under the assumption that $\mathcal{CS}$ honestly performs the intended calculation, since there is no mechanism in place to prevent or detect cheating. By computing a function, g, different than what the protocol specifies (or the intended function f but on different inputs than the legitimate ones), and using $\mathcal{SP}$ as an oracle, $\mathcal{CS}$ can learn information

about either the stored reference biometric template b_i or the fresh biometric template b'_i. As an example $\mathcal{CS}$ could compute $g(\mathsf{Enc}(b_i), \mathsf{Enc}(v))$, where v is a chosen vector by $\mathcal{CS}$, and subsequently send the result to $\mathcal{SP}$, which outputs $\mathsf{Out}_{\mathcal{SP}}$. By mounting a variant of the *hill climbing attack* [5], performing multiple repeated attempts, each time carefully choosing v, the stored template b_i can be retrieved. Such attacks against several protocols proposed in [6–8] are presented in [9–11]. Therefore, in similar applications it is important to verify the correctness of the outsourced computation, namely, the computation of XORing encrypted bitstrings. Moreover, verifiable computation of XOR is what we need in order to mitigate such an attack by a malicious $\mathcal{CS}$ on the above presented protocol. Here, we propose an efficient scheme for verifying the correctness of the outsourced XOR computation and apply it to biometric authentication. To our knowledge, the employment of verifiable computation in privacy-preserving biometric authentication has not been studied before, although the infeasibility of (fully) homomorphic encryption alone for privacy-preserving cloud computing is already known [12].

Contributions. In this work, we propose an efficient verifiable computation of XORing encrypted messages using an XOR linear message authentication code (MAC) and we build a biometric authentication protocol that is secure and privacy-preserving in the *malicious* (as opposed to the *honest-but-curious*) adversary model. In the proposed protocol, the use of homomorphic encryption (HE) and the XOR linear MAC scheme protects the privacy of biometric templates against the malicious cloud, while the secret identity to an index map provides anonymity. However, the authentication protocol does not hide access patterns from the cloud. This could be avoided using Private Information Retrieval, but at the expense of a large communication overhead. Hence we further propose an extension of the protocol using oblivious RAM (ORAM). Since $b_i \oplus b'_i$ is revealed to $\mathcal{SP}$ in the proposed protocol, we also discuss how to make it robust against leakage of information regarding the user's biometric characteristics by employing biohashing techniques.

Related work. Privacy-preserving biometric authentication has attracted considerable attention over the last decade. Multiple protocols for privacy-preserving biometric authentication are based on secure multi-party computation techniques including oblivious transfer [13] and homomorphic encryption [14, 15], as well as on private information retrieval [16, 17]. Bringer *et al.* [8] proposed a distributed biometric authentication protocol

using the Goldwasser-Micali cryptosystem [15] to protect the privacy of the biometric templates against *honest-but-curious* (or *passive*) adversaries. Nevertheless, some attacks on this protocol were reported in [5, 11, 18]. In [11], the authors have also improved upon the Bringer *et al.* protocol to achieve security against malicious but non-colluding adversaries. Simoens *et al.* [5] also presented a framework for analysing the security and privacy-preserving properties of biometric authentication protocols. In particular, they showed how biometric authentication protocols designed to be secure against *honest-but-curious* adversaries can be broken in the presence of *malicious* insider adversaries. They described several attacks against protocols proposed in [8, 18, 19]. There are also other protocols for privacy-preserving biometric authentication that are based on additive HE [14, 20] such as [21] for face recognition and its subsequent improvement in [22], as well as the protocol in [23]. Yasuda *et al.* proposed two biometric authentication protocols using somewhat HEs based on ideal lattices [6] and ring learning with errors [7], and the security of these protocols is scrutinised in [9, 10]. In most of these schemes, biometric templates are extracted as bitstrings and the similarity of two biometric templates is measured by computing the Hamming distance between them. For this reason, in [24] the authors have proposed protocols for secure Hamming distance computation based on oblivious transfer. These have potential applications in privacy-preserving biometric authentication. Recently Bringer *et al.* [25] generalised their results for secure computation of other distances such as the Euclidean and the normalised Hamming distance. Oblivious transfer was also used in SCiFi [26].

Outline. The rest of the paper is organised as follows. Sect. 2 introduces the necessary background. Section 3 presents our adversary model. In Section 4, we present our protocol for biometric authentication employing the scheme for verifiable computation of XOR. Section 5 shows how ORAM can be applied to our protocol. Finally, Section 6 concludes the paper.

2 Preliminaries

Homomorphic encryption. For our purposes, the employed HE scheme must be such that given $\mathsf{Enc}(m)$ and $\mathsf{Enc}(m')$, it is possible to homomorphically compute $\mathsf{Enc}(\mathsf{Dist}(m, m'))$, where Dist is a distance metric. We require the HE scheme to have semantic security against chosen plaintext attacks. Consider the following game played between a probabilistic polynomial time (PPT) adversary and a challenger:

$\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda)$:
$(\mathsf{pk},\mathsf{sk}), \leftarrow \mathsf{KeyGen}(\lambda)$; $(m_0,m_1),\ m_0 \neq m_1 \leftarrow \mathcal{A}(\lambda,\mathsf{pk})$;
$\beta \xleftarrow{R} \{0,1\}$; $c \leftarrow \mathsf{Enc}(m_\beta,\mathsf{pk})$; $\beta' \leftarrow \mathcal{A}(m_0,m_1,c,\mathsf{pk})$;
Return 1 if $\beta'=\beta$, 0 otherwise

and define the adversary's advantage in this game as $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda) = \left|2\Pr\left\{\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda)=1\right\}-1\right|$.

Definition 1. *We say that HE is IND-CPA-secure if all PPT adversaries have a negligible advantage in the above game:* $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda) \leqslant \mathsf{negl}(\lambda)$.

Definition 2. *A function* $\mathsf{negl}:\mathbb{N}\mapsto[0,1]$ *is called negligible if for all positive polynomials* poly *and sufficiently large* $\lambda\in\mathbb{N}$*:* $\mathsf{negl}(\lambda) < 1/\mathsf{poly}(\lambda)$.

Message authentication codes. A message authentication code (MAC) consists of $(\mathsf{KeyGen},\mathsf{TAG},\mathsf{VRFY})$ (associated with a key space, a message space and a tag space). KeyGen, a key generation algorithm, takes a security parameter λ as input and outputs a key k (i.e., $\mathsf{k}\leftarrow\mathsf{KeyGen}(\lambda)$). TAG, a tag generation algorithm, takes a message m and a key k as input, and outputs a tag (i.e., $t\leftarrow\mathsf{TAG}(m,\mathsf{k})$). VRFY, a verification algorithm, takes a message m, a tag t and a key k as input, and outputs a decision $\mathsf{Out}_{\mathsf{MAC}}$ (i.e., $\mathsf{Out}_{\mathsf{MAC}}\leftarrow\mathsf{VRFY}(m,t,\mathsf{k})$), which is 1 if the message-tag pair (m,t) is valid, and 0 otherwise.

A typical construction of a MAC is via the use of Universal$_2$ (U_2) hash functions, see [27–29] for more on U_2 hash functions. There are constructions of U_2 hash functions that are $\oplus$-linear [30], from which one can construct an $\oplus$-linear MAC scheme. Note that a MAC scheme is called $\oplus$-linear if $\mathsf{TAG}(m_1\oplus m_2,\mathsf{k}) = \mathsf{TAG}(m_1,\mathsf{k})\oplus\mathsf{TAG}(m_2,\mathsf{k})$.

Definition 3. *A MAC is called* (Q_T,Q_V,t,ϵ)*-secure (or simply* ϵ*-secure) if no PPT adversary* $\mathcal{A}$ *running in time at most* t *cannot generate a valid message-tag pair, even after making* Q_T *tag generation queries to TAG and* Q_V *verification queries to VRFY, except with probability* ϵ.

Privacy-preserving biometric authentication. A privacy-preserving biometric authentication (PPBA) protocol comprises:

- Setup: In this step, a trusted party runs the key generation algorithm KeyGen for the employed cryptographic primitives (e.g., homomorphic encryption) using a security parameter λ as input: $(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{KeyGen}(\lambda)$. The keys are distributed to the relevant parties.
- Enroll: This process collects the encrypted reference biometric template $\mathsf{Enc}(b_i)$ and stores it along with additional user information such as the user's identity ID_i in the database $\mathcal{DB}$, i.e., $\mathcal{DB}\leftarrow\mathsf{Enroll}\big(\mathsf{Enc}(b_i),\mathsf{ID}_i\big)$.

– Authen: This process takes an encrypted fresh biometric template $\mathsf{Enc}(b'_i)$ and a claimed identity ID_i, and involves actions from the protocol actors. This can be abstracted as $\mathsf{Out}_{\mathcal{SP}} \leftarrow \mathsf{Authen}(\mathsf{Enc}(b'_i), \mathsf{ID}_i)$.

The PPBA protocol is *correct* if the following definition is satisfied.

Definition 4 (Correctness). *We say that a privacy-preserving biometric authentication protocol PPBA is correct if, for all enrolled user identities ID_i with the corresponding reference biometric templates b_i, and for all fresh biometric templates b'_i, $\mathsf{Authen}(\mathsf{Enc}(b'_i), ID_i)$ results in a successful authentication of the user with ID_i if and only if $\mathsf{Dist}(b_i, b'_i) \leqslant \tau$.*

We define the security of PPBA against a malicious adversary $\mathcal{A}$ as follows. Consider the following game:

$\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda)$:
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda)$; $\mathcal{DB} \leftarrow \mathsf{Enroll}(\mathsf{ID}_i, \mathsf{Enc}(b_i))$; $b'_{i_0}, b'_{i_1}, b'_{i_0} \neq b'_{i_1} \leftarrow \mathcal{A}(\mathsf{ID}_i, \lambda, \mathsf{pk})$;
$\beta \xleftarrow{R} \{0,1\}$; $\mathsf{Out} \leftarrow \mathsf{Authen}(\mathsf{ID}_i, \mathsf{Enc}(b'_{i_\beta}))$; $\beta' \leftarrow \mathcal{A}(\mathsf{ID}_i, \lambda, \mathsf{pk}, b'_{i_0}, b'_{i_1}, \mathsf{Enc}(b'_{i_\beta}), \mathcal{DB}, \mathsf{Out})$;
Return 1 if $\beta' = \beta$, 0 otherwise

and define the adversary's advantage in this game as $Adv^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda) = \left|2\Pr\{\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda) = 1\} - 1\right|$.

Definition 5 (Security and privacy). *We say that PPBA is secure if, for all PPT adversaries $\mathcal{A}$, $Adv^{Priv}_{PPBA,\mathcal{A}}(\lambda) \leqslant \mathsf{negl}(\lambda)$.*

We assume that the adversary is given an oracle access to Authen and is allowed to query it polynomially many times, e.g., $\mathsf{poly}(\lambda)$ times, where λ may depend on the false acceptance rate. The adversary is also given $\mathsf{Enc}(b'_{i_\beta})$. If the adversary cannot distinguish whether it is $(\mathsf{ID}_i, b'_{i_0})$ or $(\mathsf{ID}_i, b'_{i_1})$ that is being used by Authen, then we say that the protocol *preserves privacy* of the biometric templates.

3 Adversary model

In this paper, we focus on *malicious* as opposed to *honest-but-curious*, adversaries and we consider a distributed setting, namely, each user $\mathcal{U}_i$ has his/her own client $\mathcal{C}_i$, a cloud computing server $\mathcal{CS}$ with its own database, and an authentication server $\mathcal{SP}$. The client $\mathcal{C}_i$ (e.g., a smartphone owned by the user $\mathcal{U}_i$) has a biometric sensor that extracts biometric templates from the user. By requiring that each user $\mathcal{U}_i$ has a client $\mathcal{C}_i$, potential damages can be minimised in case the client $\mathcal{C}_i$ is stolen or lost. We assume that each user trusts his/her own client device only to the extent that the

biometric sensor and the extracted biometric template are only accessible by the authorised apps on the user device. This is the minimal reasonable assumption given the fact that most people nowadays have a smartphone with an embedded biometric sensor, and without such a trust, users *cannot* use their devices to remotely access services. This assumption has also to be made in any type of authentication using client devices, e.g., password- or token-based remote access. This assumption does not rule out the case where an adversary is using several clients $\mathcal{C}_i$, in collusion with the cloud server, to impersonate a user that is not the owner of compromised clients. However, we do note that if a client $\mathcal{C}_i$ is compromised, say, infected by malware, then the reference biometric template of the owner $\mathcal{U}_i$ can be recovered using the fresh biometric template provided by $\mathcal{U}_i$ by hill climbing attacks [31].

The authentication server $\mathcal{SP}$ handles the keys for the employed encryption scheme and is responsible for making the authentication decision based on the underlying matching process used. We also consider the authentication server $\mathcal{SP}$ as a trusted key managing entity which keeps the secret keys secure and performs its task honestly. However, we do not trust any biometric template to $\mathcal{SP}$. The malicious party that we want to have a full protection against is the cloud server $\mathcal{CS}$. In our case the cloud has a database that stores the encrypted reference biometric templates. Additionally, $\mathcal{CS}$ performs computations on the encrypted fresh and reference biometric templates. The results of the computation will allow the authentication server to make its decision. We consider a malicious cloud server as a PPT adversary. We do not consider denial-of-service type of attacks, which are easy to mount by $\mathcal{CS}$, since it can always send a wrong response which would with high probability result in a false rejection.

Regarding communication among the protocol actors, we assume that the communication channel between the protocol entities is secure in order to avoid replay attacks. This can be achieved by using TLS or IPsec. We also only consider the case of a single client for each user, a single cloud server, and a single authentication server.

4 The scheme and the protocol

The main idea behind the verifiable computation of XOR is that the client stores homomorphically encrypted message-tag pairs (e.g., $\mathsf{Enc}(m)$, $\mathsf{Enc}(t)$, where $t = \mathsf{TAG}(m, \mathsf{k})$) in the cloud server. When the client provides a new homomorphically encrypted message-tag pair (e.g., $\mathsf{Enc}(m')$, $\mathsf{Enc}(t')$, where $t' = \mathsf{TAG}(m', \mathsf{k})$), the cloud server computes the designated func-

tion on the encrypted messages and tags separately (e.g., $\mathsf{ct}_{m\oplus m'} = f(\mathsf{Enc}(m), \mathsf{Enc}(m'))$ and $\mathsf{ct}_{t\oplus t'} = f(\mathsf{Enc}(t), \mathsf{Enc}(t'))$), and returns the results to the client. The client decrypts the results and checks if the tag is valid (i.e., $m \oplus m' \leftarrow \mathsf{Dec}(\mathsf{ct}_{m\oplus m'})$, $t \oplus t' \leftarrow \mathsf{Dec}(\mathsf{ct}_{t\oplus t'})$, and $\mathsf{VRFY}(m \oplus m', t \oplus t', \mathsf{k})$). If the MAC verification is successful, then the client can be sure (up to the security of the MAC scheme) that the cloud server has performed the correct computation.

Below, we apply this simple method to build a privacy-preserving biometric authentication protocol. In the description, HE is an encryption scheme which allows the computation of XOR of encrypted messages, i.e., $f(\mathsf{Enc}(m), \mathsf{Enc}(m')) = \mathsf{Enc}(m \oplus m')$, and MAC is an XOR linear MAC. The enrollment procedure Enroll involves the following interactions:

- $\mathcal{SP}$ generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.KeyGen}(\lambda)$ using a security parameter λ.
- The user $\mathcal{U}_i$ is asked to provide a user identity ID_i (e.g., a username or a pseudonym, etc.) by his/her client $\mathcal{C}_i$, which sends his ID_i as part of an enrollment request to $\mathcal{SP}$.
- $\mathcal{SP}$ maps ID_i to an index i (i.e., $i \leftarrow \mathsf{ID}_i$) using a secret process known only to itself. It then generates a key for the MAC using the security parameter λ and ID_i: $\mathsf{k}_i \leftarrow \mathsf{MAC.KeyGen}(\lambda, \mathsf{ID}_i)$. The tuple $(i, \mathsf{pk}, \mathsf{k}_i)$ is sent to $\mathcal{C}_i$, and pk to $\mathcal{CS}$ (the latter is only done once).
- After receiving $(i, \mathsf{pk}, \mathsf{k}_i)$, $\mathcal{C}_i$ first obtains the reference biometric template b_i from the user $\mathcal{U}_i$, computes $t_i = \mathsf{TAG}(b_i, \mathsf{k}_i)$, and encrypts the reference biometric template and the tag to obtain $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(t_i)$, respectively. $\mathcal{C}_i$ then provides $(i, \mathsf{Enc}(b_i), \mathsf{Enc}(t_i))$ to the database $\mathcal{DB}$ on the cloud server side for storage.
- $\mathcal{C}_i$ and $\mathcal{SP}$ store (i, k_i) locally.

It is important for security that the user enrollment is performed in a secure and controlled environment.

The authentication Authen involves the following interactions:

- The user $\mathcal{U}_i$ initiates the authentication process by providing his/her identity ID_i and a fresh biometric template b'_i to $\mathcal{C}_i$, which then computes $t'_i = \mathsf{TAG}(b'_i, \mathsf{k}_i)$.
- $\mathcal{C}_i$ sends ID_i as part of an authentication request to $\mathcal{SP}$, and obtains pk from $\mathcal{SP}$.
- $\mathcal{C}_i$ computes $\mathsf{Enc}(b'_i)$ and $\mathsf{Enc}(t'_i)$, and sends $(i, \mathsf{Enc}(b'_i), \mathsf{Enc}(t'_i))$ to $\mathcal{CS}$.
- $\mathcal{CS}$ retrieves $(\mathsf{Enc}(b_i), \mathsf{Enc}(t_i))$ corresp. to i from $\mathcal{DB}$ and computes $\mathsf{ct}_{b_i\oplus b'_i} = f(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i)) = \mathsf{Enc}(b_i \oplus b'_i)$ and $\mathsf{ct}_{t_i\oplus t'_i} = f(\mathsf{Enc}(t_i), \mathsf{Enc}(t'_i)) = \mathsf{Enc}(t_i \oplus t'_i)$, and sends $(\mathsf{ct}_{b_i\oplus b_i}, \mathsf{ct}_{t_i\oplus t'_i}, i')$ to $\mathcal{SP}$.

– $\mathcal{SP}$ extracts i from ID_i and checks if the extracted i and the index i' received from $\mathcal{CS}$ are equal. If $i \neq i'$, $\mathcal{SP}$ outputs $\perp$. Otherwise, $\mathcal{SP}$ retrieves the locally stored k_i corresponding to i, decrypts $\mathsf{ct}_{b_i \oplus b'_i}$ and $\mathsf{ct}_{t_i \oplus t'_i}$ to obtain $b_i \oplus b'_i$ and $t_i \oplus t'_i$, respectively. If $\mathsf{VRFY}(b_i \oplus b'_i, t_i \oplus t'_i, \mathsf{k}_i) == 0$, it outputs $\perp$. Otherwise, it checks if the Hamming weight $\mathsf{HW}(b_i \oplus b'_i) \leqslant \tau$. If this is the case, $\mathcal{SP}$ authenticates the user $\mathcal{U}_i$; otherwise, it outputs $\perp$.

From now on, we denote this protocol by PPBA-HE-MAC. It is straightforward to see that PPBA-HE-MAC is correct, since a legitimate user with his/her own legitimate device can always successfully authenticate himself/herself as long as the fresh biometric template matches the reference biometric template.

Security and privacy analysis. Intuitively, PPBA-HE-MAC is secure as long as the employed HE scheme is IND-CPA-secure (cf. Definition 1) and the MAC scheme is ϵ-secure (cf. Definition 3). In any biometric template recovery attack that makes use of the side channel information (i.e., $\mathsf{Out}_{\mathcal{SP}}$), $\mathcal{CS}$ needs to be able to submit to $\mathcal{SP}$ a $\mathsf{ct}_{b_i \oplus b'_i}$ and $\mathsf{ct}_{t_i \oplus t'_i}$ that encrypt a valid message-tag pair. The ϵ-security of the employed MAC scheme does not allow this to happen. Furthermore, if $\mathsf{Out}_{\mathcal{SP}} == \perp$, $\mathcal{CS}$ does not know whether it is due to the MAC verification failure or the mismatch between the fresh and the reference biometric template. Hence, the protocol is secure against the malicious $\mathcal{CS}$. The following summarises the security of our protocol, and the proof is given in Appendix-A.

Theorem 1 (Security and privacy). *The protocol PPBA-HE-MAC is secure and privacy-preserving against the malicious $\mathcal{CS}$ according to our Definition 5, if the employed HE is IND-CPA-secure and MAC ϵ-secure.*

Simulation. PPBA-HE-MAC is efficient because both the MAC scheme and the HE scheme can be implemented efficiently. The efficiency of the $\oplus$-linear MAC scheme in our case depends on the efficiency of the employed U_2 hash functions. One suitable family of U_2 hash functions for our instantiation is the construction by Krawczyk [30], which exploits a Linear Feedback Shift Register to allow efficient hardware implementations. This construction is also efficient on software. We refer the curious reader to [32] for more on the software performance of U_2 hash functions.

Note that our utilisation of a lightweight MAC scheme for verifying the correctness of the outsourced computation contrasts nicely with the

existing verifiable computation schemes. More precisely, efficiency is the main issue with the existing verifiable computation schemes since they are very heavy computationally and have a large overhead [33]. On the other hand, our approach using a MAC scheme is very efficient regarding computation cost.

Regarding the HE scheme, we demonstrate its efficiency by simulating the Goldwasser-Micali encryption scheme [15] for various security levels and biometric template lengths. The Goldwasser-Micali encryption scheme supports homomorphic evaluation of the XOR operation, and their primitives are the most heavy ones in our construction.

The simulations were performed on a Intel®Core™2 Duo CPU E8400 @ 3.00GHz x2 64 bit CentOS Linux 7 computer. The simulation software, written in C++, linked the NTL v9.4.0 (Number Theory Library [34]), GNU Multiple Precision Arithmetic Library v6.0.0 [35], for efficient multi-precision arithmetics support. The security level and the corresponding size of the prime factors are chosen according to the ECRYPT II recommendations and the length of the biometric binary templates is chosen following Daugman [36] and SCiFI [26]. The simulation setup and results are shown in Table 1, the source code can be provided upon request via anonymous channels.

Table 1: Simulation setup and results for the Goldwasser-Micali scheme.

Security level in bits	Size of prime factors in bits	Binary biometric template length	Mean template encoding time $[s]$	Mean template decoding time $[s]$
80	1248	900	$9.22 \cdot 10^{-3}$	$2.06 \cdot 10^{-1}$
		2048	$2.09 \cdot 10^{-2}$	$4.69 \cdot 10^{-1}$
128	3248	900	$3.79 \cdot 10^{-2}$	$6.51 \cdot 10^{-1}$
		2048	$8.60 \cdot 10^{-2}$	1.48

We remark that since our aim is to show the feasibility of the HE scheme, the implementation is not optimised. Also, the simulations are run on single core, even though the Goldwasser-Micali encryption and decryption procedures can be done in parallel, since it is a bitwise encryption scheme. Therefore, the simulation results show that the HE scheme required for our instantiation is not only feasible, but also efficient.

5 Protocol extensions

Oblivious RAM (ORAM) for hiding access patterns. Our protocol can be easily extended to protect the access pattern of the client $\mathcal{C}_i$ towards the cloud server $\mathcal{CS}$. However, existing methods such as Private

Information Retrieval (PIR) come at an elevated communication overhead. To reduce such costs, we suggest the use of ORAM instead, as a more suitable mechanism, and its use, as presented by this work, would not alter the underlying security properties of the main protocol. ORAM allows a client to hide the entry as well as the access pattern from the server at a significantly reduced communication vs PIR. Moreover, ORAM security is derived from the indistinguishability of any two access patterns $A(y)$ and $A(y')$, for any two respective queries y and y'. The concept was initially presented by Goldreich and Ostrovsky [37] in 1996. Since then, the field has seen the introduction of various protocols with improved mechanisms and primitives, e.g., [38]. These advances on protocol efficiency have motivated the apparition of new applications such as, biometric identification [39]. Typically, ORAMs are designed and used to solve the problem of $\mathcal{DB}$ outsourcing [40]. This model would require the user to execute various ORAM primitives so that the remote database is correctly shuffled. To alleviate this processing task, and to make our protocol user agnostic, we propose to use a Secure Multiparty Computation (MPC) scheme. MPC schemes have been suggested in combination with ORAM constructions in recent works (e.g., [41]).Under this extended protocol, every time a new user data (e.g., $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(t_i)$) is added to the ORAM $\mathcal{DB}$. The index i is used to store the data mapping in a separate ORAM. The following are the additional parties, operations and the protocol extension:

- $\mathcal{MPC}$ **Agent:** MPC mechanisms provide security against semi-honest or malicious adversaries and in various coalitions, including computational security against dishonest majorities e.g., [42]. An $\mathcal{MPC}$ agent, composed by different distrustful players (computational parties) with competing interests can be added to our scheme. These computational parties can be as many as needed, to give the users confidence on the scheme and could be allocated by any combination of the scheme participants. This agent has to store, in shared form, an ORAM containing the mapping of the template database using i.
- **MAP**(i)**:** It returns the mapping of the template based on the shared index i from the user. The mapping corresponds to the position to be queried on the remote ORAM $\mathcal{DB}$ template.
- **Sh**(i)**:** It is used to represent the secure secret sharing of the index i.
- **Enrollment:** The enrollment procedure is the same as described in Section 4. However, at the end of the scheme, the client $\mathcal{C}_i$ provides $(\mathsf{Sh}(i), \mathsf{Enc}(b_i), \mathsf{Enc}(t_i))$ to the $\mathcal{MPC}$ agent, who then stores i on its local mapping ORAM and appends $\mathsf{Enc}(b_i), \mathsf{Enc}(t_i))$ to the $\top$ position of the physical $\mathcal{DB}$ of the cloud ORAM.
- **Authentication:** Similarly to the Enrollment, the authentication procedure follows the same steps that are described at Section 4. In the same spirit as before, once the client $\mathcal{C}_i$ has computed $(\mathsf{Enc}(b_i'), \mathsf{Enc}(t_i'))$, it is sent to the

$\mathcal{MPC}$ agent instead, together with the stored index i in shared form. Then, the agent uses i to extract the template and grants access to the cloud storage, so that the original process can continue. To avoid revealing i to the $\mathcal{CS}$, the $\mathcal{MPC}$ agent sends the index directly towards the $\mathcal{SP}$ as i'.

These protocol extensions are oriented towards a task distribution. Hence, they do not have an impact on the security properties of the authentication scheme. It is worth noticing, however, that the security with respect to the access pattern will depend solely on the underlying ORAM and MPC protocols used by any implementation.

Biohashing for avoiding linkability of error patterns. The error pattern $b_i \oplus b'_i$ is disclosed to $\mathcal{SP}$ at the end of the authentication phase, as shown in Section 4. This can disclose some information about the binary biometric templates. For instance, the reliability of each bit can be different among different users, so the error patterns can be used for tracking users. In the ideal case, all the error patterns should be equiprobable for all the users. In this case, disclosing the error patterns would not provide any advantage to $\mathcal{SP}$. However, this is difficult to achieve in practice.

A practical solution to this problem is to use biohashing techniques [43]. The usual approach for obtaining binary templates b_i from biometric features f_i is by using a user-independent binarization transformation $b_i = B(f_i)$. Biohashing consists of using a user-specific random transformation $b_i = B_i(f_i)$ instead. The specific design of these transformations ensures a minimum distortion in the distances in the transformed domain with respect to the distances in the original domain, thus keeping the discrimination ability of the biometrics unaffected. And the dependency between the error patterns and the user-specific binary templates' reliability is avoided, since changing B_i leads to an independent error pattern.

The incorporation of biohashing into our system is straightforward. The user-specific random transformation B_i is generated during the enrollment phase in the user client $\mathcal{C}_i$, where it is stored and used to obtain the enrollment binary template $b_i = B_i(f_i)$. During the authentication phase, this transformation is used by $\mathcal{C}_i$ to obtain $b'_i = B_i(f'_i)$. When the user enrolls again, a new random transformation would be generated, thus avoiding linkability between the previous and the new error patterns.

6 Conclusions

We proposed an efficient scheme for verifiable computation of XORing encrypted messages, and successfully applied it to the scenario of distributed biometric authentication, where the storage of the encrypted biometric templates and part of the computations are outsourced to a cloud server. The security and privacy of the proposed scheme has been proved in a challenging and reasonable malicious internal adversarial scenario, as opposed to the more usual and less realistic honest-but-curious scenario. Additionally, ORAM is employed instead of prevalent PIR schemes to reduce the communication overhead while keeping the access pattern hidden from the cloud. Moreover, Biohashing techniques are proposed to avoid the disclosure of linkable error patterns. The efficiency of the proposed scheme has been assessed by simulating the most computationally costly parts of the proposed scheme, i.e. the homomorphic encryption primitives, showing the feasibility and efficiency of the proposed solution.

Acknowledgments. This work was funded by the European Commission through the FP7 project "EKSISTENZ," with grant number: 607049. This work was also partially supported by the FP7-STREP project "BEAT: Biometric Evaluation and Testing", grant number: 284989 and the VR project PRECIS

References

1. Costello, C., Fournet, C., Howell, J., Kohlweiss, M., Kreuter, B., Naehrig, M., Parno, B., Zahur, S.: Geppetto: Versatile verifiable computation. In: IEEE S&P, IEEE (2015) 253–270
2. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: CRYPTO 2010. LNCS (2010) 465–482
3. Zhang, L.F., Safavi-Naini, R.: Batch verifiable computation of outsourced functions. Designs, Codes and Cryptography (2015) 1–23
4. IriTech Inc.: Irisecureid: Cloud-based iris recognition solution. http://www.iritech.com/products/solutions/cloud-based-iris-recognition-solution-0 (2016) Accessed: 2016-05-18.
5. Simoens, K., Bringer, J., Chabanne, H., Seys, S.: A framework for analyzing template security and privacy in biometric authentication systems. IEEE Transactions on Information Forensics and Security **7**(2) (2012) 833–841
6. Yasuda *et al.*, M.: Packed homomorphic encryption based on ideal lattices and its application to biometrics. In: Security Engineering and Intelligence Informatics. Volume 8128 of LNCS. (2013) 55–74
7. Yasuda *et al.*, M.: Practical packing method in somewhat homomorphic encryption. In: DPM/SETOP. Volume 8147 of LNCS. (2013) 34–50

8. Bringer, J., Chabanne, H., Izabachène, M., Pointcheval, D., Tang, Q., Zimmer, S.: An application of the Goldwasser-Micali cryptosystem to biometric authentication. In: ACISP 2007. Volume 4586 of LNCS., Springer (2007) 96–106
9. Abidin, A., Mitrokotsa, A.: Security aspects of privacy-preserving biometric authentication based on ideal lattices and ring-lwe. In: Proceedings of the IEEE Workshop on Information Forensics and Security. (2014) 1653–1658
10. Abidin, A., Pagnin, E., Mitrokotsa, A.: Attacks on privacy-preserving biometric authentication. In: NordSec 2014. Volume 8788 of LNCS., Springer (2014) 293–294
11. Abidin, A., Matsuura, K., Mitrokotsa, A.: Security of a privacy-preserving biometric authentication protocol revisited. In: CANS 2014. Volume 8813 of LNCS., Springer (2014) 290–304
12. Van Dijk, M., Juels, A.: On the impossibility of cryptography alone for privacy-preserving cloud computing. In: Proceedings of the 5th USENIX Conference on Hot Topics in Security. HotSec'10, USENIX Association (2010) 1–8
13. Yao, A.C.C.: How to generate and exchange secrets. In: Foundations of Computer Science, 1986., 27th Annual Symposium on, IEEE (1986) 162–167
14. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: EUROCRYPT 1999. Volume 1592 of LNCS. (1999) 223–238
15. Goldwasser, S., Micali, S.: Probabilistic encryption & how to play mental poker keeping secret all partial information. In: Proceedings of the fourteenth annual ACM symposium on Theory of computing. STOC 1982, ACM (1982) 365–377
16. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. Journal of the ACM **45**(6) (1998) 965–981
17. Ostrovsky, R., Willian E. Skeith, I.: A survey of single-database private information retrieval: techniques and applications. In: PKC'07. LNCS, Springer (2007) 393–411
18. Barbosa, M., Brouard, T., Cauchie, S., de Sousa, S.M.: Secure biometric authentication with improved accuracy. In: ACISP 2008. Volume 5107 of LNCS., Springer (2008) 21–36
19. Stoianov, A.: Security issues of biometric encryption. In: Proceedings of the 2009 IEEE Toronto International Conference on Science and Technology for Hunanity (TIC- STH). (September 2009) 34–39
20. Damgård, I., Geisler, M., Krøigaard: Efficient and secure comparison for on-line auctions. In: ACISP 2007. Volume 4586 of LNCS., Springer (2007) 416–430
21. Erkin, Z., Franz, M., Guajardo, J., Katzenbeisser, S., Lagendijk, I., Toft, T.: Privacy-preserving face recognition. In: PETS 2009. (2009) 235–253
22. Sadeghi, A.R., Schneider, T., Wehrenberg, I.: Efficient privacy-preserving face recognition. In: ICISC 2009. LNCS (2009) 229–244
23. Huang, Y., Malka, L., Evans, D., Katz, J.: Efficient privacy-preserving biometric identification. In: NDSS. (2011)
24. Bringer, J., Chabanne, H., Patey, A.: SHADE: Secure hamming distance computation from oblivious transfer. In: Financial Cryptography Workshops. (2013) 164–176
25. Bringer, J., Chabanne, H., Favre, M., Patey, A., Schneider, T., Zohner, M.: GSHADE: Faster Privacy-preserving Distance Computation and Biometric Identification. In: Proceedings of the 2nd ACM Workshop on Information Hiding and Multimedia Security, ACM (2014) 187–198
26. Osadchy, M., Pinkas, B., Jarrous, A., Moskovich, B.: SCiFI - A System for Secure Face Identification. In: IEEE S&P 2010. (May 2010) 239–254
27. Carter, L., Wegman, M.N.: Universal classes of hash functions. J. Comput. Syst. Sci. **18** (1979) 143–154

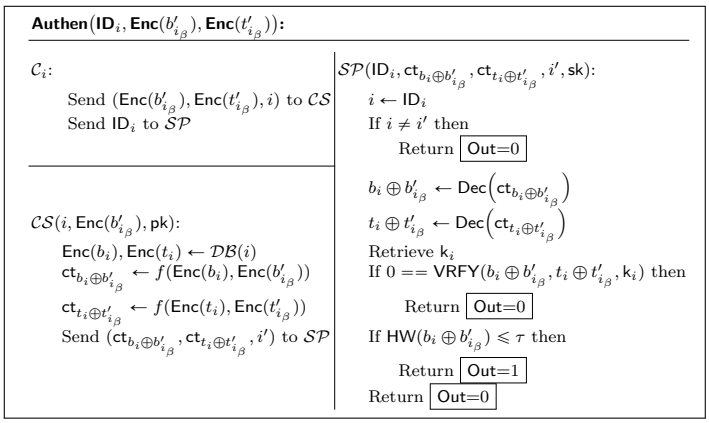
28. Stinson, D.R.: Universal hashing and authentication codes. In Feigenbaum, J., ed.: CRYPTO '91. Volume 576 of Lecture Notes in Computer Science., Springer 1992 (1991) 74–85
29. Abidin, A., Larsson, J.Å.: New universal hash functions. In Lucks, S., Armknecht, F., eds.: WEWoRC 2011. Volume 7242 of LNCS., Springer (2012) 99–108
30. Krawczyk, H.: Lfsr-based hashing and authentication. In Desmedt, Y., ed.: CRYPTO '94. Volume 839 of Lecture Notes in Computer Science., Springer 1994 (1994) 129–139
31. Pagnin, E., Dimitrakakis, C., Abidin, A., Mitrokotsa, A.: On the leakage of information in biometric authentication. In: INDOCRYPT 2014. Volume 8885 of LNCS., Springer (2014) 265–280
32. Nevelsteen, W., Preneel, B.: Software performance of universal hash functions. In: EUROCRYPT'99. LNCS, Springer (1999) 24–41
33. Walfish, M., Blumberg, A.J.: Verifying computations without reexecuting them. Commun. ACM **58**(2) (2015) 74–84
34. Shoup, V.: NTL: A library for doing number theory. `http://www.shoup.net/ntl/` (2016) Accessed: 2016-02-26.
35. GMP: The GNU Multiple Precision Arithmetic Library. `https://gmplib.org/` (2016) Accessed: 2016-02-26.
36. Daugman, J.: How iris recognition works. In: ICIP (1). (2002) 33–36
37. Goldreich, O., Ostrovsky, R.: Software protection and simulation on oblivious rams. J. ACM **43**(3) (May 1996) 431–473
38. Faber, S., Jarecki, S., Kentros, S., Wei, B.: Three-Party ORAM for Secure Computation. In: ASIACRYPT 2015. Springer (2015) 360–385
39. Bringer, J., Chabanne, H., Patey, A.: Practical identification with encrypted biometric data using oblivious ram. In: ICB 2013. (2013) 1–8
40. Karvelas, N., Peter, A., Katzenbeisser, S., Tews, E., Hamacher, K.: Privacy-preserving whole genome sequence processing through proxy-aided oram. In: WPES '14, ACM (2014) 1–10
41. Keller, M., Scholl, P.: Efficient, Oblivious Data Structures for MPC. In: ASIACRYPT 2014. Springer (2014) 506–525
42. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: CRYPTO 2012. Volume 7417 of LNCS., Springer (2012) 643–662
43. Teoh, A.B.J., Yuang, C.T.: Cancelable biometrics realization with multispace random projections. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **37**(5) (2007) 1096–1106

A Proof of Theorem 1

Proof. Let Π be the PPBA-HE-MAC protocol. The security of Π against a malicious adversary $\mathcal{A}$ (i.e., $\mathcal{CS}$) is defined via the following game.

$\mathsf{Exp}^{\mathsf{Priv}}_{\Pi,\mathcal{A}}(\lambda, \mathsf{ID}_i)$:
$(\mathsf{pk}, \mathsf{sk})$, k_i, $\mathsf{MAC}.K \leftarrow \mathsf{KeyGen}(\lambda, \mathsf{ID}_i)$; $\quad \mathcal{DB} \leftarrow \mathsf{Enroll}(\mathsf{ID}_i, \mathsf{Enc}(b_i), \mathsf{k}_i)$
(b'_{i_0}, b'_{i_1}), $b'_{i_0} \neq b'_{i_1} \leftarrow \mathcal{A}(\mathsf{ID}_i, \lambda, \mathsf{pk}, \mathsf{MAC}.K)$;
$\beta \xleftarrow{R} \{0,1\}$; $\quad t'_{i_\beta} \leftarrow \mathsf{TAG}(b'_{i_\beta}, \mathsf{k}_i)$; $\quad \mathsf{Out} \leftarrow \mathsf{Authen}(\mathsf{ID}_i, \mathsf{Enc}(b'_{i_\beta}), \mathsf{Enc}(t'_{i_\beta}))$;
$\beta' \leftarrow \mathcal{A}(\mathsf{ID}_i, \lambda, \mathsf{pk}, b'_{i_0}, b'_{i_1}, \mathsf{Enc}(b'_{i_\beta}), \mathsf{Enc}(t'_{i_\beta}), \mathcal{DB}, \mathsf{Out})$;
Return 1 if $\beta' = \beta$, 0 otherwise

where $\mathsf{MAC}.K$ is the key space for the employed MAC. The adversary's advantage is defined as $\mathsf{Adv}^{\mathsf{Priv}}_{\Pi,\mathcal{A}} = \left|2\Pr\{\mathsf{Exp}^{\mathsf{Priv}}_{\Pi,\mathcal{A}}(\lambda, \mathsf{ID}_i) = 1\} - 1\right|$. If the advantage is $\leqslant \mathsf{negl}(\lambda)$, we say that Π is secure (and preserves the privacy of biometric templates) against $\mathcal{A}$.
The details of $\mathsf{Authen}\big(\mathsf{ID}_i, \mathsf{Enc}(b'_{i_\beta}), \mathsf{Enc}(t'_{i_\beta})\big)$ are given below.



The proof is based on the following two hybrid games. **game** 0: This is the original game. Let S_0 be the event that $\beta' = \beta$.
game 1: This is the same as **game** 0, except that now $\mathcal{CS}$ always performs the correct computation. Let S_1 be the event that $\beta' = \beta$ in **game** 1.

Since providing a different index i' than the correct one i always results in $\perp$ output, it does not help the adversary (i.e., the cloud) to win any of the games. So we assume that $\mathcal{CS}$ always provides the correct index i.
Claim 1: $|\Pr\{S_0\} - \Pr\{S_1\}|$ is negligible. This follows from the ϵ-security of the MAC scheme. Precisely, the difference between the two games is that in **game** 0, $\mathsf{VRFY}(b_i \oplus b'_{i_\beta}, t_i \oplus t'_{i_\beta}, \mathsf{k}_i) == 0$ if $\mathcal{CS}$ does not perform the computation correctly, except for probability ϵ, while in **game** 1, that does not happen as it performs the computation correctly. So the difference between the winning probabilities in **game** 0 and **game** 1 is negligible.
Claim 2: The adversary has negligible advantage in **game** 1, i.e., $\left|2\Pr\{S_1\} - 1\right| \leqslant \mathsf{negl}(\lambda)$. This follows from the IND-CPA-security of the employed HE scheme. Since otherwise, we can use the adversary $\mathcal{A}$ as a blackbox to construct another PPT adversary $\mathcal{A}'$ that can win the IND-CPA game against the HE scheme with non-negligible probability in a straightforward fashion. More precisely, the adversary $\mathcal{A}'$ can use the challenge ciphertext in the IND-CPA game to simulate the Π for $\mathcal{A}$, and use $\mathcal{A}$'s guess to win the IND-CPA game against the HE scheme. Hence, combining the two claims, we have that $\mathsf{Adv}^{\mathsf{Priv}}_{\Pi,\mathcal{A}}$ is negligible.